Master's thesis

Master's Programme in Data Science

Information Criteria and Effective Feature Size Estimation for Data with Inherent Dependencies

Ioanna Bouri

May 20, 2019

Supervisor(s): Associate Professor Teemu Roos

Examiner(s): Associate Professor Teemu Roos
Janne Leppä-Aho

University of Helsinki
Faculty of Science

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Degree programme
Faculty of Science	Master's Programme in Data Science

Tekijä — Författare — Author

Ioanna Bouri

Työn nimi — Arbetets titel — Title

Information Criteria and Effective Feature Size Estimation for Data with Inherent Dependencies

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidantal — Number of pages
Master's thesis	May 20, 2019	37

Tiivistelmä — Referat — Abstract

In model selection, it is necessary to select a model from a set of candidate models based on some observed data. The model should fit the data well, but without being overly complex, since that would not allow the model to generalize well its predictions to unseen data. Information criteria are widely used model selection methods that select a model based on some criteria. Information criteria estimate a score for each candidate model, and use that score to make a selection. A common way of estimating such a score, rewards the candidate model for its goodness of fit on some observed data and penalizes for the model complexity.

Many popular information criteria, such as Akaike's Information Criterion (AIC) and Bayesian Information Criterion (BIC) penalize model complexity by the feature dimension. However, in a non-standard setting with inherent dependencies, these criteria are prone to over-penalizing the complexity of the model.

Motivated by how these commonly used criteria tend to over-penalize, we evaluate AIC and BIC on a multi-target setting with correlated features. We compare AIC and BIC, with the Fisher Information Criterion (FIC), a criterion that takes into consideration correlations amongst features and does not penalize model complexity solely by the feature dimension of the candidate model.

We evaluate the feature selection and predictive performances of the three information criteria in a linear regression setting with correlated features. We evaluate the precision, recall and F1 score of the set of features each criterion selects, compared to the feature set of the generative model. Under this setting's assumptions, we find that FIC yields the best results, compared to AIC and BIC, both in the feature selection and predictive performance evaluation.

Finally, using FIC's properties for feature selection, we derive a formulation that allows to approximate the effective feature dimension of models with correlated features, in linear regression settings.

ACM Computing Classification System (CCS):
Theory of Computation → Machine Learning Theory

Avainsanat — Nyckelord — Keywords

machine learning, model selection, information criteria, feature selection, multi-target predictions

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Preface

This thesis is a product of two years of research, thanks to Prof. Teemu Roos who gave me the opportunity to study with him, collect research and teaching experience already during my Master's years. Equally I would like to thank PhD student Janne Leppä-Aho, who guided me throughout this research with great insight and ideas on the experimental work. Finally, I would personally like to thank from my heart my parents, George and Xena, none of this would be achieved without their love and encouragement.

Notation

The notation used throughout this thesis is consistent for the following variables:

Table 1: Notation

Variable	Explanation
x	instance vector for multi-target setting
$\mathscr{X}$	set of instance vectors
t	target vector for multi-target setting
$\mathscr{T}$	set of target vectors
(x_i, t_j)	dyad of the i-th instance and j-th target
y_{ij}	response variable for dyad (x_i, t_j)
Y	set of response variables
n	sample size
d	number of free parameters of model
β	coefficient vector
ε	standard normal noise
X	feature array of dimension $n \times d$
$\hat{L}$	maximized likelihood estimate
M	degrees of freedom in polynomial
$\hat{\sigma}_n^2$	residual variance of submodel for sample size n
${\tilde{\sigma}_n}^2$	residual variance of full model for sample size n
σ^2	residual variance of model when $n \to \infty$
λ	regularization parameter for LASSO
K	number of iterations in experiment

Contents

1. Introduction

There is an infinite number of models that can fit to a collection of sample data. However, it is a challenge to approximate the generative distribution of a set of data. Model selection holds a very crucial role in machine learning theory, since it is used to estimate the most suitable model from a set of candidate models, that approaches best the true model that has generated the sample data.

Most popular ways to perform model selection are cross-validation techniques and a variety of information criteria that take different aspects into consideration as to point out the optimal model for an application setting. Most available literature on those methods investigates standard settings with data that lacks strong dependencies.

What is interesting in this analysis, is that frequently used information criteria, such as Akaike's Information Criterion (AIC) and Bayesian Information Criterion (BIC) penalize the complexity of the model solely by its number of free parameters. Therefore, AIC and BIC would be prone to over-penalizing complexity in settings where the data is generated based on parameters that are not independent. For this reason, the main research question is to evaluate the performance of AIC and BIC on a linear regression setting with correlated features. We choose to simulate a dyadic prediction setting, since it is suitable for having responses that are dependent on multiple variables. We include in the analyses the Fisher Information Criterion (FIC) which estimates model complexity in a more suitable manner for a setting with correlations. FIC tries to estimate the number of features that contain useful information based on the observed data. We present an evaluation and comparison of all three criteria, using the dyadic prediction setting.

We investigate the dyadic prediction setting, where the data encodes strong inherent dependencies. We simulate the data generation for such a setting, and analyze the behavior of each information criterion with different levels of correlation in the setting. Furthermore, we evaluate the feature selection and predicitve performances of each information criterion, in comparison to the generative model of the data. Finally, we present a formulation of FIC that can be used to approximate the effective feature dimension for a setting with correlated features, in linear regression settings.

2. Dyadic Setting

In this chapter, we discuss the setting we are working with. First, we discuss the multi-target structure of the setting, then an example application of drug-target interactions with dyadic data. Finally, we present how we model the dyadic data linearly.

Multi-target prediction

In standard supervised learning, the task is to predict a single target variable based on a set of features. However, according to the definition given by Waegeman et al. (2019), multi-target prediction problems focus on predicting multiple target variables at the same time. These multi-target variables do not have to be of the same type, but can be depending on the application.

A multi-target setting consists of instances $x \in \mathscr{X}$ and targets $t \in \mathscr{T}$. In some applications, there is some available information, called side information, that is not included in the input space or the output space, but can be useful in the learning task. A multi-target setting with no side information available for the instances x and the targets t is called conventional. Dyadic prediction is a special case of multi-target prediction, when some side information about target relations is available. In the case of the dyadic setting, the response for each dyad (x_i, t_j) will be the respective y_{ij}. Therefore, each data point in the training set will be a triplet of the form (x_i, t_j, y_{ij}). In such a setting, the predictive task at hand is given the dyad (x_i, t_j), to try and predict the response variable y_{ij}. Specifically in the linear regression setting we assume, x_i represents a feature vector that describes the i-th instance and, respectively, t_j is the feature vector that descibes the j-th target.

We consider a dyadic prediction for this thesis, since it provides a setting where responses are dependent on a multiple number of variables, which facilitates the development of strong dependencies amongst the data. In the dyadic prediction setting used in this thesis, we assume that we have available side information about target relations and instance relations as well.

The instance-target dyads (x_i, t_j) can describe a pair of variables of different types. This means that the instances and the targets can be any combination of categorical

or numerical data. The type of the response variable Y also defines the learning setting. When Y is a categorical variable, then we have a multi-target classification problem, while when Y is numerical the setting is a multi-target regression.

According to Pahikkala et al. (2014a), four experimental settings arise based on whether the dyad to be predicted consists of elements that have been encountered during the training of the prediction model. Assuming we have a dyad $(x_i, t_j) \in \mathscr{X} \times \mathscr{T}$ and we seek to predict its response y_{ij}:

1. Both x_i and t_j have been encountered in the training input to the prediction model.
2. x_i has been encountered in the training input to the prediction model, but t_j was not included in the training data. Estimating y_{ij} in this case can be done by taking into consideration side information, like similarities amongst targets.
3. t_j has been encountered in the training input to the prediction model, but x_i was not included in the training data. Same as above, estimating y_{ij} in this case can be done by taking into consideration side information, like similarities amongst instances.
4. None of the dyad elements (x_i, t_j) have been encountered during the training phase. This setting is non-trivial, and the prediction depends solely on the affinities amongst instances and amongst targets.

From these four experimental settings, setting 1 is the closest to a standard supervised learning scenario, as all dyads to be predicted were included in the training input of the prediction model. The settings 2 and 3 are more common settings, where only part of the dyad has been encountered during the training phase and the model can predict using either similarities amongst instances or similarities amongst the targets as side information. Setting 4 is the most challenging, as the dyad whose response variable needs to be predicted has not been encountered before during the training phase of the prediction model. In this last case, the prediction has to depend both on similarities with the other instances and similarities with the other targets. The problem that arises with this last setting is known as the full cold start problem in dyadic prediction, as presented for instance in Pahikkala et al. (2014b).

Multi-target predictions have many applications in different fields, and are the basic framework for many settings, particularly useful in image tagging, computer vision, drug design, but also in other domains like ecology, biology, chemistry, where responses are dependent on a multiple number of variables. In the next section, we present an example drug design application that follows the dyadic setting.

Drug-target example application

As we previously discussed, a multi-target prediction setting has a vast variety of applications, particularly in problems where predictions must be made conditioning on multiple dependent features. Following the work of Pahikkala et al. (2014a), we discuss an example drug design application to which the dyadic setting applies.

In this setting, the $x \in \mathscr{X}$ represent the feature vectors describing drug instances and respectively $t \in \mathscr{T}$ represent the respective drug targets. As shown in Figure 2.1, the response variable Y encodes the interaction level between drug (rows) and target (columns), and takes numerical values with range 0 - 10000, 10000 representing the lowest interaction level, and 0 being the highest level of interaction. All of the four experimental multi-target settings are possible depending on the drug-target dyads (x_i, t_j) that we are learning. More specifically, this drug design setting focuses on target-based drug discovery, where each drug compound targets selectively a particular protein target. Figure 2.1 illustrates the cases where the drugs (setting 2), or the targets (setting 3), or both (setting 4) have not been observed during the training phase. The grey rows and columns represent these data that were unseen during the training. In setting 4, we can see an illustration of the cold-start program since it can be very challenging to predict the value of the response variable y_{ij} on the intersection of an unseen row x_i and column t_j.

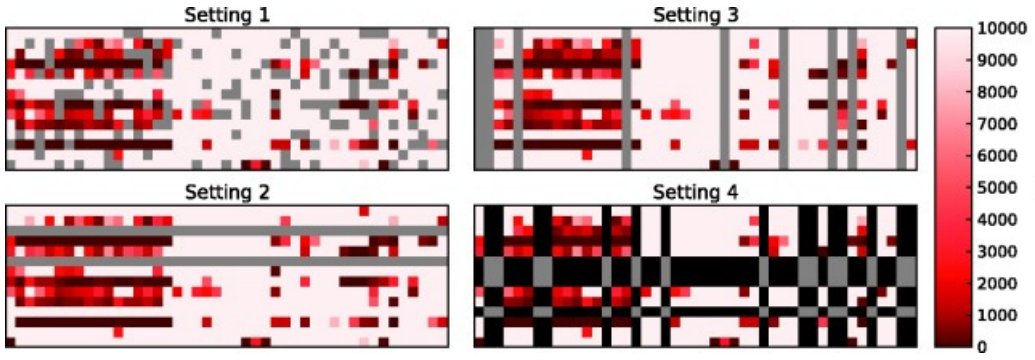


Figure 2.1: The four experimental settings of multi-target predictions depending on whether the elements of the dyad were included in the training input data of the prediction model. The response variable y_{ij} shown here is indicative of the level of interaction between the members of the dyad (x_i, t_j) (Figure from Pahikkala et al. (2014a)). In this example, the level of interaction ranges between 0-10000, where zero encodes the strongest level of interaction and 10000 the lowest.

Most machine learning approaches on drug-target interaction prediction treat the problem as binary classification, predicting whether there will be an interaction or not, by assuming some interactivity threshold. However, it is important to encode how tightly the drug compound binds to a particular protein kinase, since this depends on numerous factors. Therefore, it is interesting to investigate this problem as a multi-target regression setting, looking into the level of the drug-target interaction affinity.

Drug design problems like the one described, help speed up the experimental work necessary for developing a new drug. Such machine learning settings opt to prioritize different compound combinations that are most likely to yield the desired results in drug development. Furthermore, such settings can help in discovering more about existing drug compounds. Machine learning applications on such settings can learn structural similarities between drug compounds, as well as genomic similarities between the drug targets, and this enables us to predict new targets for existing drug compounds.

In the drug-target setting there is a number of different drug compounds that are highly similar or actually repeated. Repetitions in the setting's instances translate into similarities amongst the features. These inherent dependencies make the meaningful statistical information encoded in the feature set different to, for instance, the statistical information contained in a model with independent features where the whole feature set would be as informative. This property of dyadic settings motivates us to use them for evaluating the feature selection performance of different information criteria.

In the next section, we present the modeling process of a dyadic setting using simple linear regression, while we preserve the key property of repeating instances. We imitate the drug repetitions presented in this example application, as to create strong dependencies amongst the data.

Modeling dyadic data linearly

By modeling the data as dyads with repetitions of the instances $x \in \mathscr{X}$, we achieve a setting with strong dependencies. The motivation to model such a setting, is to evaluate the feature selection and predictive performances of models selected by different information criteria on correlated data. Due to the challenges arising from modeling a non-standard setting, we decide to make some assumptions that simplify the setting. In this thesis, when selecting how to model a dyadic setting, we focus on the feature selection performance of the information criteria, and not on the predictive performance of the models. Therefore, we try to use as simple models as possible, even if they do not yield the best predictive performances. We decide to model the dyadic setting using simple linear regression as to associate the dyads (x_i, t_j) with its respective response y_{ij}.

Next we take a look at the general case of linear regression and later we discuss how we use it to simulate a dyadic setting. Linear regression models linearly the relationship between one or more independent variables to the response variable. In the case of the dyadic setting, it models linearly each dyad (x_i, t_j) with their respective responses y_{ij}. The general case of simple linear regression can be expressed by modeling the response variable y_i as:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_d x_{id} + \varepsilon_i, \text{ with } i = 1, 2, ..., n \ .$$

These equations for n data points and d features, are equal to the matrix notation:

$$y = X\beta + \varepsilon \ ,$$

where,

$$y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix},$$

$$X = \begin{pmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \dots & x_{1d} \\ 1 & x_{21} & \dots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \dots & x_{nd} \end{pmatrix},$$

$$\beta = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_d \end{pmatrix} \text{ and } \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$

The response variable Y can be generated linearly by adding some standard normal noise ε to the inner product of the coefficient vector β with the feature matrix X. When we fit a linear regression model, we basically estimate the coefficient vector that fits the training input data of the prediction model. Essentially, each feature's coefficient encodes the amount of influence a particular feature has in generating a particular value for the response variable.

In the proccess of keeping the learning setting as simple as possible, we make the assumption that the components of the dyad are generated independently from one another.

Therefore, we can stack them together and consider a simple linear regression setting. We also assume that the instance feature vectors and target feature vectors are of the same dimension d. This essentially means that, under these assumptions, there is one design matrix X with dimension $n \times 2d$, that includes the instance features and the target features for each dyad. From this point on, we refer solely to this single feature vector X, that includes the features x_i and t_j of both components of each dyad. To summarize, the dyad's instance component is repeated, whereas the targets are randomly distributed and different for each dyad encountered in the training input. Figure 2.2 gives an illustration of how the model's features are structured:

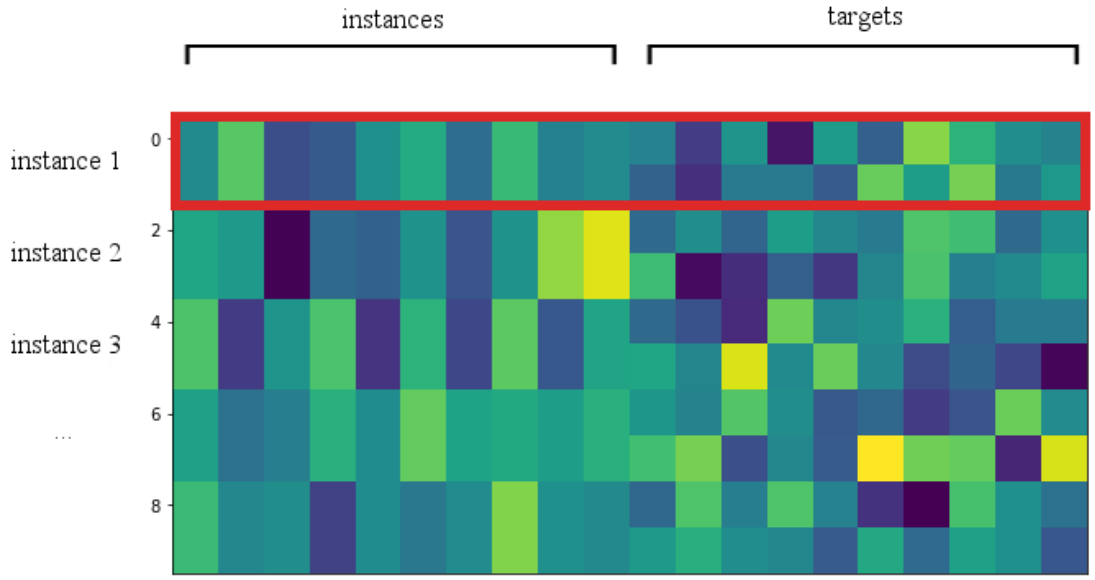


Figure 2.2: An example of the dyadic feature representation. Here, each instance is repeated twice, while their respective targets are independent. For example, here instance 1 is represented by the first two dyads (x_1, t_1) and (x_1, t_2), that are shown in the red frame. The more repetitions of dyads with the same instance, the stronger the dependencies amongst the features.

Feature selection is the process that seeks the most informative combination of features, and enables us to eliminate redundant features. In order to evaluate the information criteria, we need to model a setting such that not all of the features contribute to the generation of the response variable Y. We need to include redundant features that the information criteria should optimally be able to detect and disregard in their complexity estimation, instead of penalizing over the dimension of the full model which includes feature repetitions. For this purpose, we use feature selection to produce different submodels of a full model. A full model contains all features including possible repetitions, whereas the submodels consist of different subsets of these features.

In order to seek for the most informative submodels, some features of the full model need to contribute more than others when the responses are generated. This can be done in a controlled manner so that each information criterion's selected submodel can be compared to the true generative model.

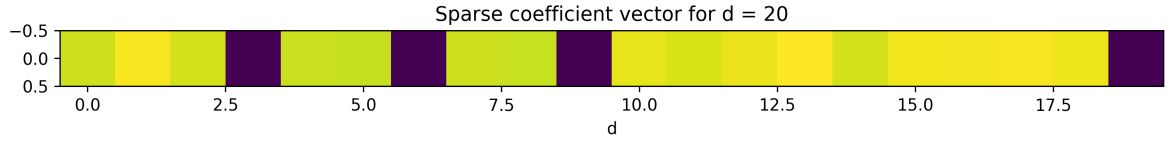


Figure 2.3: An example of a sparse coefficient vector with $d = 20$ and 20% sparsity. Sparse elements are represented in purple, therefore these are the coefficients that deactivate the respective feature vector elements.

As illustrated in Figure 2.3, we choose to use a sparse coefficient vector so that only a percentage of the original full feature set has an effect in the produced responses. For instance, a coefficient vector for feature dimension $d = 100$ with a 20% sparsity (equivalent to 80% density), will have 20 elements in random indices in the coefficient vector as zeros. This coefficient vector is then multiplied by the feature matrix in linear regression. Therefore the sparse elements of the vector will deactivate the features in the corresponding indices of the feature vector. This essentially means that these deactivated features do not influence the generation of the corresponding responses. This way, when we start the feature selection process, we know that the ideal submodel is the one with the 80 features that actually influence the responses, and we are ablte to see that the 20 features in the indices of the sparse elements in the coefficient vector do not influence the predictive performance of the model.

3. Feature Selection

In this chapter, we review feature selection in theory and how it is linked to model selection. We discuss model selection with information criteria, introducing shortly the information criteria we investigate in our experiments: AIC, BIC and the FIC. Furthermore, we include a few words on LASSO since we use it largely in our experiments. Finally, we discuss shortly cross-validation since it is a very powerful model selection tool.

Model selection background theory

According to the theory presented in Grünwald (2007), model selection is the process of choosing the optimal model amongst a set of candidate models, based on specified criteria and given some observed data. As Occam's razor admonishes us, amongst a set of candidate models with similar predictive performances, the simplest option is the best. The purpose of model selection is to seek an equilibrium between how well each candidate model fits the given data versus how complex it is, conditioning on the criteria specified.

For instance, we consider the case of a polynomial regression as illustrated in Fig. 3.1 from Bishop (2006). The green curve represents the generative function $sin(2\pi x)$ that produces the data shown as blue dots. The red curve in each subplot shows different polynomial fittings on the same data. What changes in each plot is the degree of freedom M for each polynomial fitting. For example, in the first plot the degree is zero, which means a constant function. Next, for $M = 1$, we have a straight line but with a slope that follows how the particular data is distributed. In the next case, with $M = 3$ we have a fit that follows the nature of the data, and yet seems to be able to generalize to more data from the same generative function. Finally, in the last case we have a 9th degree polynomial, which has fitted a curve that describes each particular training point. Although the last case might be the best fit on this particular training data, it results in a very complex model and overfits the data in such way that it makes it very hard to generalize on other unseen data from the original generative function. The optimal model could vary depending on particular criteria used for model selection, but graphically this illustration seems to support the case for degree of freedom $M = 3$ where there is a good balance between goodness of fit yet the model seems able to generalize well on data

outside this specific training data set.

The above example justifies the equilibrium between goodness of fit and model complexity that model selection seeks to satisfy. Redundant number of features means an unecessarily complex model, which can lead to overfitting the data.

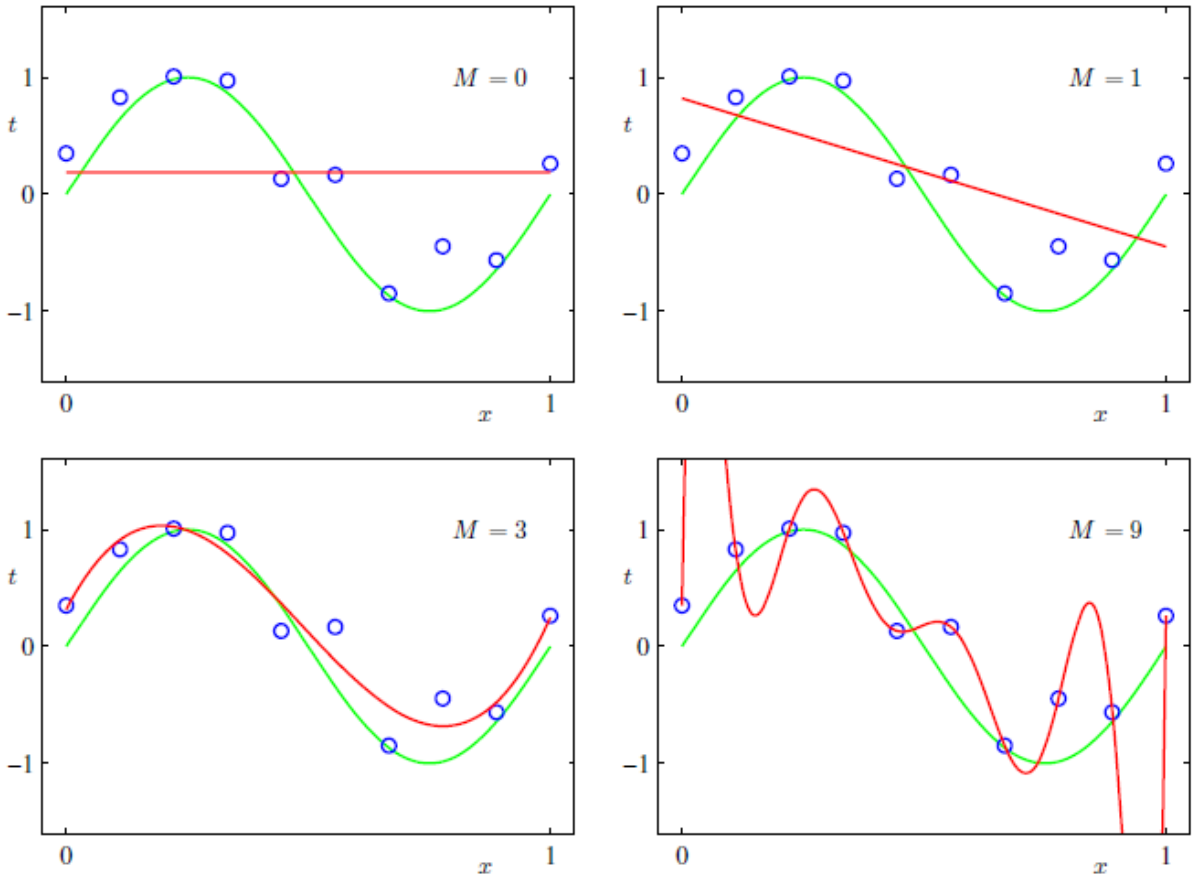


Figure 3.1: An illustration of the goodness of fit and model complexity tradeoff (Figure from Bishop (2006)).

Feature selection focuses on methods for selecting a subset of features of a model, that can yield a satisfying predictive performance, while making the model less complex. It is very closely related to model selection, since different candidate subsets of features essentially result in different models, each with a subset of the features. This thesis is motivated by the fact that in the more popular information criteria, model complexity is penalized solely by the model's number of free parameters. However, in the case of a setting with dependencies, this can be a misleading measure of model complexity. For instance, if there are feature repetitions, using feature selection can result in an informative submodel that does not include repetitions of the same feature. This submodel can consist of a subset of the features and be used instead of the full model, with equivalent predictive performances.

Feature selection is important since reducing the number of features, leads to simpler models that can generalize better on unseen data. Moreover, the training becomes computationally cheaper and faster. We can use feature selection to produce different subset variations of the full set of features. This way we build all possible submodels that can be produced by different combinations of the features of a full model. A way to use information criteria for feature selection, is to perform model selection with some criterion on these candidate submodels, and evaluate which submodel was selected by each criterion.

There is a vast variety of different model selection techniques using different criteria for choosing the optimal model. Most of them work by ranking all candidate models by a certain score evaluating their suitability given the specified criteria. Information criteria and cross-validation are most commonly used model selection methods. In the next section, we review information criteria, and present in more detail some of the most popular ones.

Information criteria

Information Criteria are estimators that evaluate which model is the optimal selection from a set of candidate models, based on a predefined set of criteria and some observed data. There is a variety of such estimators available and they usually consist of formulas that measure the goodness of fit of a model for an observed set of data and penalize for the complexity of the model. Very commonly used information criteria include AIC and BIC, as well as numerous variations of those. In this section, we shortly discuss some background theory for information criteria, as well as review AIC and BIC. We also present the FIC, which we compare to the commonly used AIC and BIC on the correlated dyadic setting we presented in the previous chapter.

When fitting a model, we opt to approximate the generative distribution of some observed data, while trying to maintain the ability to generalize well over unseen data. Therefore, models can only approximate real settings, which means they will always lead to some loss of information. Information criteria make use of particular heuristics to evaluate this loss for each model of a set of candidate models, and suggest the most suitable model that would minimize that loss based on the specified criterion.

There is a vast variety of information criteria available, each using a different method to estimate goodness of fit and the complexity of each model. Goodness of fit can be measured by evaluating the model likelihood on observed data, whereas there are different ways to penalize for model complexity based on the number of free parameters of the model and the sample size of the observed data. Often, the information criteria scores are derived by subtracting the likelihood estimation from the model complexity term, with minimum score being the best for the candidate models.

Commonly used information criteria include AIC, also the corrected Akaike's Information Criterion (AICc) and BIC. A very thorough analysis of popular information criteria can be found in Emiliano et al. (2014). We compare these criteria to the FIC. FIC has very interesting properties for an experimental setting with inherent dependencies like the drug-target setting we introduced. FIC's key property is that it does not penalize model complexity strictly by the number of parameters like AIC and BIC do, but also takes into consideration dependencies amongst each model's features.

In our experiments, we investigate how AIC, BIC and FIC work with correlated data, however there are other information criteria alternatives. Some examples of other approaches include Subspace Information Criterion (SIC) presented by Sugiyama and Ogawa (2001), the Kernel-based Information Criterion (KIC) introduced by Danafar et al. (2014) and kernel-based Information Complexity (ICOMP) presented by Zhang (2007). In the next section, we shortly present AIC, BIC and FIC.

Akaike's Information Criterion

AIC was introduced by statistician Hirotugu Akaike (1974). AIC estimates the quality of each model given a set of candidate models and a set of observed data. When a model seeks to approximate the generative distribution of the observed data, it will almost never be exact. Therefore, some of the original information is lost through the model's representation of the generative process.

AIC score represents the relative amount of this information loss. The smaller the score is, i.e. the less information loss, the better the score for the model. From a set of candidate models, the model that yields the smallest information loss will be selected as the optimal option. The AIC formulation consists of a term representing the goodness of fit on the particular observed data and the model complexity term. Finally, the AIC score estimation is based on a trade-off between these two terms. The more complex a model is, the better it can fit the given data making it prone to overfitting and vice versa (respectively underfitting the data), (Sakamoto et al., 1986).

$$AIC = 2d - 2\log(\hat{L})$$

This is the AIC formulation, where d is the number of free parameters of the model, and $\hat{L}$ is the maximized likelihood estimate for the model fitting the observed data. For each candidate model, the AIC score is calculated with the same set of observed data and the model with the minimum AIC score is considered the best option. The complexity term here is $2d$ and penalizes the goodness of fit, which here is the log-likelihood term $2\log(\hat{L})$ of the model fitting the observed data.

The linear regression setting assumes that the error terms are normally distributed, and this allows for the maximized likelihood to be expressed in terms of residual variance. Since we assume a linear regression setting, we find it useful to present the resulting derivation of the maximized likelihood and AIC, which we also use in our computations (see for instance Rawlings et al. (1998)). The log-likelihood, in terms of residual variance, can be expressed as:

$$\log(\hat{L}) = -\frac{n}{2}\log(\hat{\sigma}_n^2) + C \ , \tag{3.1}$$

where C is a constant that depends only on the particular observed data and $\hat{\sigma}_n^2$ is the residual variance with:

$$\hat{\sigma}_n^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \ ,$$

where y_i is the true value of the response variable Y for the i-th data point and $\hat{y}_i$ is its predicted value.

Finally, using the above formulas, we can derive the following AIC formulation in terms of residual variance:

$$AIC = 2d + n \log(\hat{\sigma}_n^2) - 2C \ .$$

However, since C is the same for a particular data set and information criteria evaluate different models always on the same data set, we can omit the last term:

$$AIC = 2d + n \log(\hat{\sigma}_n^2) \ .$$

From the AIC formulation, it also becomes evident that as the size of observed data n tends to infinity, the log-likelihood term of AIC gets more important than the penalty term.

The reason we include AIC in our experiments is to compare an information criterion that is widely used, yet indifferent to dependencies amongst the features. AIC penalizes for the full number of parameters of a model, regardless of any similarities amongst them. Therefore, we expect AIC to over-penalize for model complexity in the case of settings with dependencies.

Bayesian Information Criterion

BIC was formulated by Gideon E. Schwarz (1978). BIC is very closely related to AIC. A BIC score is estimated for each model in the set of candidate models with their likelihood penalized by the complexity term. In the following BIC formulation, d is the number of free parameters of the model and n is the sample size of the observed data. As in the AIC formulation, the log-likelihood term remains the same with $2log(\hat{L})$:

$$BIC = \log(n)d - 2\log(\hat{L}) \ .$$

Same as in AIC, the BIC penalizes complexity for the number of parameters of the model, the only difference being that the penalty term in BIC is larger than AIC. Again, the model with the minimum BIC score is the optimal selection based on BIC.

Similarly to AIC, and under the assumption of a linear regression setting, we can use the maximized likelihood formulation (3.1) to express BIC in terms of residual variance:

$$BIC = n \log(\hat{\sigma}_n^2) + d \log(n) \ .$$

The main difference to AIC is the complexity term which instead of being $2d$, here it is $\log(n)d$. This means that the complexity term is essentially larger than in AIC, and dependent on the sample size n. From the formulations of AIC and BIC, it becomes evident that both criteria penalize model complexity based on the increase of the number of features. From the point that a more complex model cannot improve significantly the predictive performance, redundant features only cause the model to overfit the observed data. This motivates why we seek to investigate further FIC, which does not penalize model complexity solely by the number of parameters of each model, like AIC and BIC do.

Fisher Information Criterion

In this section, we discuss the FIC in more detail. We show how it is different to the more popular information criteria we have seen so far. First, we review some basic background theory about the formulation of FIC. Next, we discuss how it might hold an advantage to the formerly introduced information criteria, particularly in a setting with inherent dependencies like the dyadic setting used in our experimental work.

FIC background theory

AIC and BIC penalize model complexity solely by the number of parameters of each model. This can lead to over-penalizing model complexity in the case of a setting with strong dependencies. However commonly used both AIC and BIC are, this is a possible caveat they can present in non-standard settings. Therefore, we investigate the FIC that penalizes complexity by the Fisher Information term , which represents the dimensionality of the informative features (Fisher, 1950).

The FIC was primarily introduced by Wei (1992), as an extension of the Predictive Least Squares criterion (PLS) developed by Rissanen (1986). Our motivation for choosing to include FIC in the information criteria we are comparing, is that FIC's complexity estimation does not depend coarsely on the number of parameters of the model. Instead, the FIC complexity term is formulated by the conditional Fisher information matrix (Ly et al., 2017). This term has been selected proportionally to the logarithm of the statistical information contained in a model using the full set of its features.

The FIC score for a model M is given in the following formulation:

$$FIC(M) = n\hat{\sigma}_n^2 + \tilde{\sigma}_n^2 \log\det\left(\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i'\right).$$

FIC is a criterion primarily introduced for feature selection, therefore considers a full model (FM) and a sumbodel (M) that consists of a subset of FM's features. The variances

$\hat{\sigma}_n^2$ and ${\tilde{\sigma}_n}^2$ correspond to the residual variances of the submodel M and the full model FM, respectively. Furthermore, n represents the sample size of the observed data, and $\mathbf{x}_i$ is the feature vector for each observed data point in the submodel M.

Here the term $\log\det\left(\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i'\right)$ estimates model complexity. However, the criterion penalizes in a different manner than the previously reviewed AIC and BIC. FIC measures complexity differently than AIC and BIC. Instead of penalizing by the number of parameters, the model complexity is estimated as the logarithm of the determinant of the conditional Fisher Information matrix for the feature vector of the setting. The determinant of this term essentially represents the amount of useful information encoded in the features of the model. In specific cases, FIC can be linked to BIC. According to Wei (1992), under some assumptions, FIC can be asymptotically equivalent to PLS. In specific settings, we can write:

$$n\log\left(\mathrm{PLS/n}\right) = n\log\left(\hat{\sigma}_n^2\right) + d\log\left(n\right)(1+o(1)) \ ,$$

and when $n\to\infty$ the term $o(1)\to 0$, which results in:

$$n\log\left(\mathrm{PLS/n}\right) \underset{n\to\infty}{=} n\log\left(\hat{\sigma}_n^2\right) + d\log\left(n\right) \ . \tag{3.2}$$

In linear regression settings, this formula is essentially the same with BIC expressed in terms of residual variance. From this we can expect $n\log(\mathrm{FIC}/n)$ to have a similar behavior with BIC, when $n\to\infty$. Therefore, we can use this transformation $n\log(\mathrm{FIC}/n)$ to present all three criteria in a comparable scale.

Considering a simple linear regression setting, Figure 3.2 illustrates how the FIC penalty estimator behaves over increasing level of correlation amongst the features. We generate features with different level of correlation for each measurement, and generate linearly some responses. We use sample size $n = 4000$, $d = 10$ and we generate each design matrix using different levels of positive correlation in the range $0.1 - 0.9$. For each model we evaluate the term $\log\det\left(\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i'\right)$ of FIC. We observe that FIC will penalize less as the correlation amongst the features increases. It is evident from the formulations of AIC and BIC, that they are expected to have the same penalty regardless of the level of correlation amongst the features.

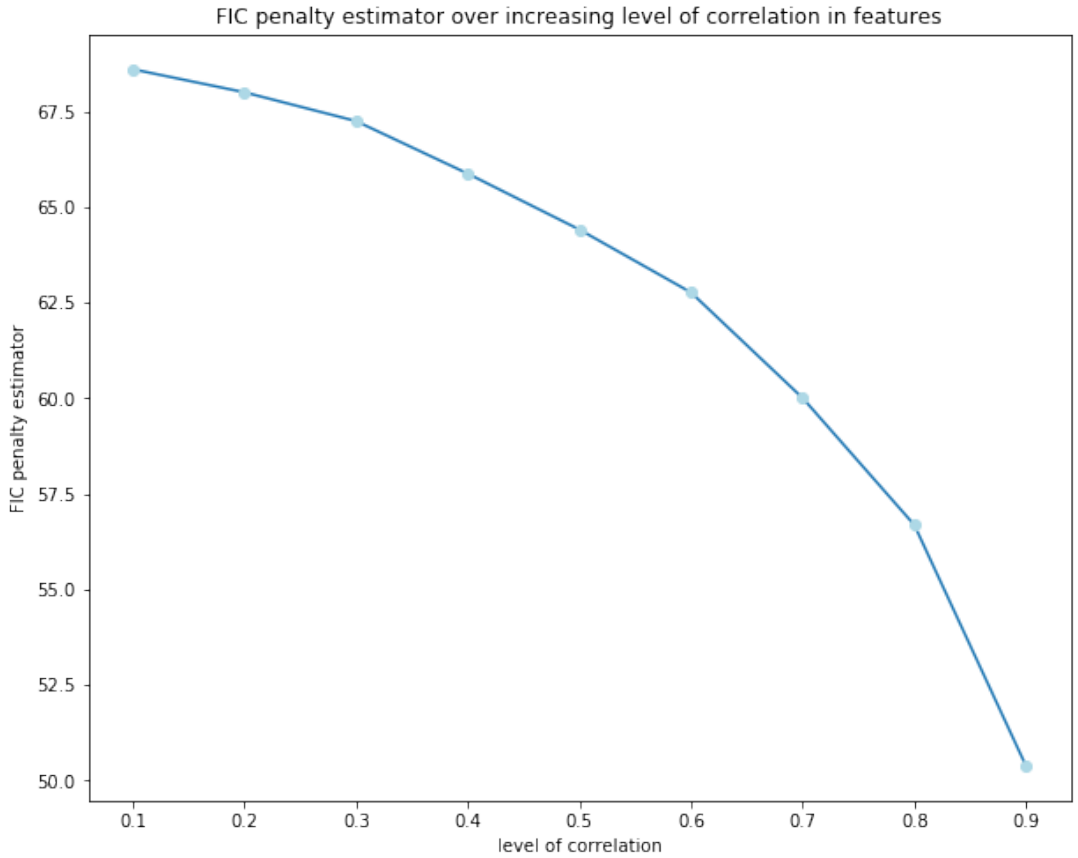


Figure 3.2: This Figure illustrates how the FIC penalty estimator behaves over increasing level of correlation amongst the features. We generate features with different level of correlation for each measurement, and generate linearly some responses. We use sample size $n = 4000$, $d = 10$ and we generate each design matrix using different levels of positive correlation in the range $0.1 - 0.9$. For each model we evaluate the term $\log\det\left(\sum_{i=1}^{n} \mathbf{x}_i\mathbf{x}_i'\right)$ of FIC.

Application on the dyadic setting

In contrast to AIC and BIC, FIC penalizes for model complexity without depending solely on the number of free parameters of the model. The crterion penalizes more for model complexity when the term $\log\det\left(\sum_{i=1}^{n} \mathbf{x}_i\mathbf{x}_i'\right)$ is estimated for a feature that is uncorrelated to the rest of the features and has a large magnitude. This property holds since features with larger magnitude, have a proportionally large share of responsibility for the produced prediction errors.

For a submodel M, if a subset of its features bears strong similarities, then FIC will consider penalizing for only one feature, instead for all its repetitions as well. However, AIC and BIC are expected to penalize for the full number of parameters, disregarding any possible similarities amongst the features.

For example, if we estimate the information criteria scores of a model with 5 different

features, plus one repeated feature, we essentially have a total number of features $d = 6$ for the full model. AIC and BIC are expected to penalize for the full number of parameters, whereas FIC is expected to penalize only by the number of informative features. Here, since one instance out of the 6 is duplicate, the informative features would be $d' = 5$.

The above example, leads to the conclusion that AIC and BIC are prone to overpenalizing models with dependencies in their features. However, FIC can adapt to settings with correlated features as well as to the independent cases, considering no additional penalty for redundant, uninformative features.

These properties of the FIC complexity term, motivate us to compare it with AIC and BIC in our experiments. We investigate the behavior of the three criteria in a setting with dependencies like the dyadic setting introduced in Chapter 2.

In the next section we briefly introduce the concept of feature selection using l_1-norm regularized linear regression, known as the Least Absolute Shrinkage and Selection Operator (LASSO). We present LASSO, since it is an important tool that we include in our experimental work and use to produce different submodels from the same full model.

Feature selection with LASSO

In this section, we briefly present l_1-norm regularized linear regression, also known as Least Absolute Shrinkage and Selection Operator (LASSO). We explain how it works and why it is useful in this application. Although it is not the central theme in our theoretical investigation, it is, nevertheless, an important tool in our experimental work.

LASSO background theory

LASSO is a widely used regularization technique. As presented by Tibshirani (1996), LASSO is a regularized regression that learns to fit a less complex model to the training data, as to avoid overfitting. LASSO essentially uses l_1-norm regularization, thus choosing the less important features of a model and setting them to zero. This is a property that works very well for reducing the feature dimension of too complex models by deactivating the k least important features. l_1-norm regularized linear regression means adding a regularization term to the linear regression loss function. Using least squares to evaluate the loss, LASSO's objective function can be formulated as:

$$L(X, \beta, \lambda) = \frac{1}{2n}(y - X\beta)^2 + \lambda\|\beta\|_1 \ .$$

In the above formulation, y represents the true responses, whereas $X\beta$ are the estimated responses from the linear regression model. λ is the regularization parameter, with some $\lambda > 0$. $||\cdot||_1$ denotes the l_1-norm. The term $\frac{1}{n}(y - X\beta)^2$ is the mean squared error (MSE). We try to optimize the LASSO fit by minimizing the objective loss function. Here MSE is further scaled by $\frac{1}{2}$ to cancel out the '2' that the derivative of the exponent of the squared error produces.

Formally, LASSO for different λ values, produces different sparsity levels of the coefficient vector β. Intuitively, larger λ values lead to sparser solutions for β. Specifically, from a certain λ value and higher, the coefficient vector will essentially consist solely of zeros.

Application on the dyadic setting

The FIC was primarily introduced for feature selection. Our main research question involves estimating the three information criteria we have reviewed so far, for different models consisting of correlated features. Since FIC considers a full model and compares the scores of its different submodels, we simulate the original data by generating the correlated features and responses for the full model.

The next step is to try different combinations of features in order to create different submodels of the full model. Different greedy heuristics can be used to achieve different combinations and subsets of the full model's features. We use LASSO to efficiently produce these submodels.

In practice, we fit a LASSO regression model with a range of different λ values. For each λ value, we obtain a coefficient vector β with a different sparsity level. The higher the value of λ, the sparser the coefficient vector will be. LASSO's heuristics provide us with a sequence of submodels from the most complex models that may contain more redundant information, to the sparsest solution possible that is an empty model. An empty model essentially means a coefficient vector of zeros that does not allow any of the full model's features to influence the response estimate.

Once we obtain the sparse coefficient vector for a specified λ value, it is then useful for deactivating the feature vectors that correspond to the indices that are filled with zeros in the coefficient vector. With this technique, we manage to obtain different subsets of the full model's features, and therefore different submodels for which we can estimate AIC, BIC and FIC. Finally, our focus is to see which of these submodels, or respective λ values, are selected as the optimal model by each criterion.

Cross-validation

Although the concept of cross-validation somewhat differs from what we have been discussing so far about information criteria, cross-validation offers a very powerful feature selection tool. Cross-validation can in some cases be computationally challenging, but, nevertheless, it is a very widely used model selection method.

Cross-validation formally is a model validation technique, used to obtain a confident estimate about the model's predictive performance. The use of cross-validation allows the comparison of the predictive performances of different candidate models, estimated on common grounds.

There are many different types of cross-validation, but we review the traditional concept that makes it a useful model selection tool. The partitioning of the data into training and validation subsets, can always result in different performance estimates for the model, depending on the partitioning of the data and the sample sizes used. Cross-validation can give more confident results by repeating a number of different partitioning rounds on the data and repeating the predictive performance estimation for the different partitionings. Traditional k-fold cross-validation involves splitting the data in k such subsets, training the model every time with k-1 subsets of data and evaluating the model on the k-th subset. This process is repeated k times, so that we obtain an error estimate for all k subsets. The partitionings are traditionally done randomly, but other partitioning techniques may be more suitable depending on the application. For instance, stratified splits amongst data that represent different classes are very useful for classification tasks where the class representations are imbalanced in the available data. Finally, cross-validation averages over the different repetitions to reach a more confident estimate of the model's predictive performance.

Cross-validation is a powerful feature selection method, as well. One way to do feature selection with cross-validation, involves a process that begins with no predictors in the model, and then an estimate of the cross-validation error of adding each predictor. Next, we choose to add the predictor that yields the smaller cross-validation error, and continue this process until the inclusion of any of the remaining predictors does not give a statistically significant improvement.

From the above description, it becomes rather clear that it can be more computationally challenging than information criteria. Information criteria estimate the goodness of fit of each model by its log-likelihood and no repetitive procedure is required to estimate the predictive performance of each model. For instance, as Burnham and Anderson (2002) explain, with the traditional k-fold cross-validation there are k splits and fittings required, which become more and more computationally challenging by increasing the sample size n. However, increasing the sample size n means it is less prone to overfitting

with more training data. The change of sample size n influences computationally the information criteria estimation very little, compared to cross-validation.

Regardless the computational differences, information criteria are closely linked to cross-validation techniques. As Stone (1977) showed, under some assumptions, AIC is asymptotically equivalent to leave-one-out-cross-validation (LOOCV). Nevertheless, cross-validation is a very powerful tool in model validation and selection. It is also interesting to see how cross-validation could apply on a setting with correlations, as to maintain the information about data dependencies in each partition and repetition of the process. Other tools such as bootstrap and stratified splits can be useful to maintain the balance of dependent data in every cross-validation partition.

4. Experimental Results

In this chapter we present the results of our experimental work. In the first section, we summarize the generation process of the dyadic setting, as well as the process we follow in our main experiments. We begin the experiments with a motivating example of effective feature dimension estimation. We present a method for estimating the effective feature dimension for a model with correlated features using a FIC formulation. Next, we produce submodels from a full generative model and estimate the AIC, BIC and FIC scores for each of these submodels as to compare the selected submodel of each criterion to the true model. Finally, we compare the submodels that each criterion selected by their feature selection and predictive performances.

Experimental process overview

In this section, we discuss the generation of the dyadic setting and present an overview of the process and parameterization we follow in our main experiments. We simulate the production of different submodels by assuming a full model that consists of all their features. We begin by generating the full model features and later pick subsets of those features using LASSO.

The first step is to generate the features of the full model. For simplicity, we assume that the instance and target feature vectors are of the same dimension d. Since instance features and target features are generated independently from one another, we can stack them together and consider a simple linear regression setting. Therefore, the feature array is of size $n \times 2d$, where there are d instance attributes and d target attributes describing each sample dyad. The instance attributes are repeated multiple times with different target attributes to encode strong dependencies amongst the data. In the proccess of keeping the learning setting as simple as possible, we begin by generating the features from a standard normal distribution with mean 0 and variance 1.

We then generate the interaction responses linearly. We need to generate a true model that does not use all its features to produce the linear responses. This way, when we produce different submodels consisting of subsets of the features of the full model, there are elligible candidates that each criterion can select as an optimal selection. In order to achieve that, we generate from a standard normal distribution a sparse coefficient vector as seen in Figure 2.3. The zero elements of the sparse coefficient vector, will essentially deactivate the features in the corresponding indices of the full model's design matrix. Using this technique, the linear responses depend only on a subset of the features, which the information criteria try to estimate by selecting the submodel of the informative features. Finally, to produce the responses linearly we add some standard normal noise to the product of the features with each coefficient vector.

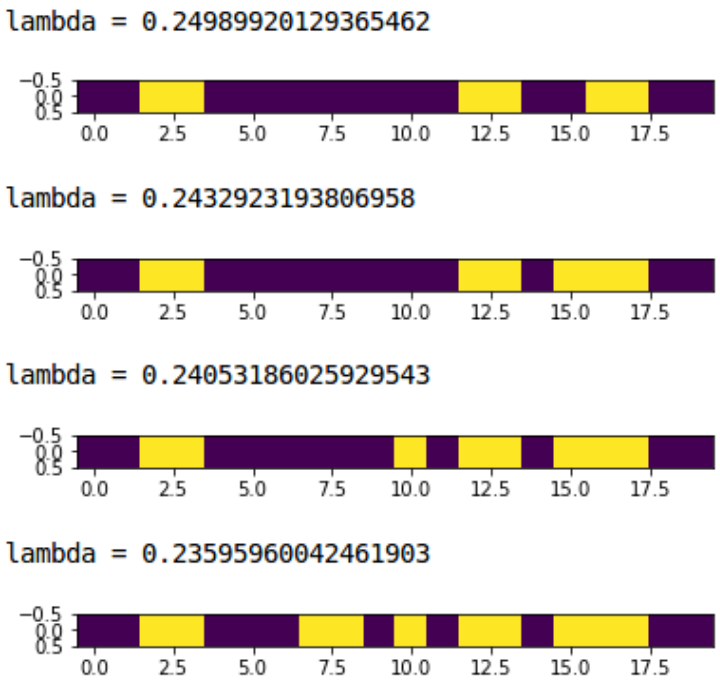


Figure 4.1: Generating submodels with correlated features from the full model with LASSO. For each λ value, a submodel with different sparsity levels is produced. The purple indices represent zero elements of the coefficient vector, whereas the yellow show the location of non-zero elements. The sparse coefficient vector cancels out the features in the indices of zero elements. In the following experiments, we produce submodels using 15 different λ values uniformly distributed in the range $(0, 1)$.

We produce different coefficient vectors with varying sparsity levels. Each coefficient vector multiplied by the full feature vector can then produce a different submodel of the full model. For this purpose, we use l_1-norm regularized regression (LASSO) to generate these submodels. Using different λ values gives us different levels of sparsity of the coefficient vector of the full model. The larger the value of λ, the sparser the produced submodel. In the following experiments, we produce submodels using 15 different λ

values uniformly distributed in the range $(0, 1)$. As Figure 4.1 illustrates, when λ is decreasing, the coefficient vectors generated become less sparse. The purple indicates the zero elements of the coefficient vectors. Each coefficient vector produces each alternative submodel by deactivating different subsets of features based on the indices of the zero elements.

Below in Figure 4.2, we show an overview of the steps and parameterizations that we follow to simulate the dyadic setting designed for submodel selection. In the resulting setting, we can estimate AIC, BIC and FIC scores on the different submodels and conduct our experiments as shown in the figure below.

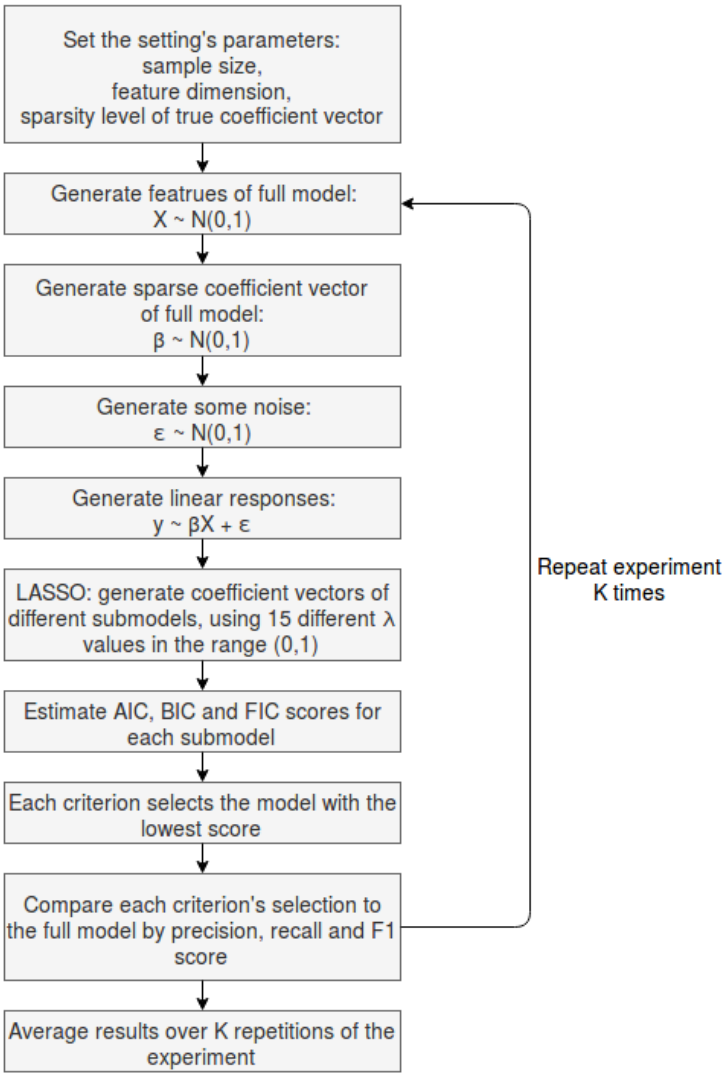


Figure 4.2: Overview of experiment steps.

Effective feature size estimation

Motivated by the feature selection properties of FIC, we derive a formulation of the information criterion for estimating the effective feature dimension of a model with correlated features.

In order to extract the feature dimension of a model using the BIC formula, we can subtract the goodness of fit term from the BIC score and solve for the feature dimension d:

$$\text{BIC} - n \log(\hat{\sigma}_n^2) = d \log(n)$$

$$d = \frac{\text{BIC} - n \log(\hat{\sigma}_n^2)}{\log(n)} \ . \tag{4.1}$$

Following the derivation of 3.2, we presented a link between $n \log(\text{FIC/n})$ and BIC. Therefore, we can derive d^* in the same way for $n \log(\text{FIC/n})$ as we did for BIC:

$$d^* = \frac{n(\log(\text{FIC/n}) - \log(\hat{\sigma}_n^2))}{\log n} \ . \tag{4.2}$$

Therefore, the feature dimension d^* can be estimated in terms of the FIC score of a model.

FIC score does not penalize model complexity by its full dimension, but only for the number of features that have a proportionally large contribution to the produced prediction errors. Therefore, the derived formula 4.2 used with the FIC score can approximate an effective feature dimension d^* for a model with correlated features. However, if the formula 4.1 that is derived from BIC, is used on a model with correlated features, it will only calculate the full dimension of the model d. The derivation of 4.2 can help with feature selection in an efficient manner. After approximating an effective dimension for some full model with correlated features, the submodel selection can focus on combinations of features for models of this estimated dimension, narrowing down the number of candidate submodels significantly.

An example implementation follows, where we simulate a setting of feature dimension $d = 10$ where 5 of these features describe different targets for the same instance. This means that the effective number of parameters for an optimal model is less than the full dimension of the full model, when the features are correlated. In this example, the useful parameters are 5 for the different instances plus one more describing the rest of the 5 samples that share the same instance.

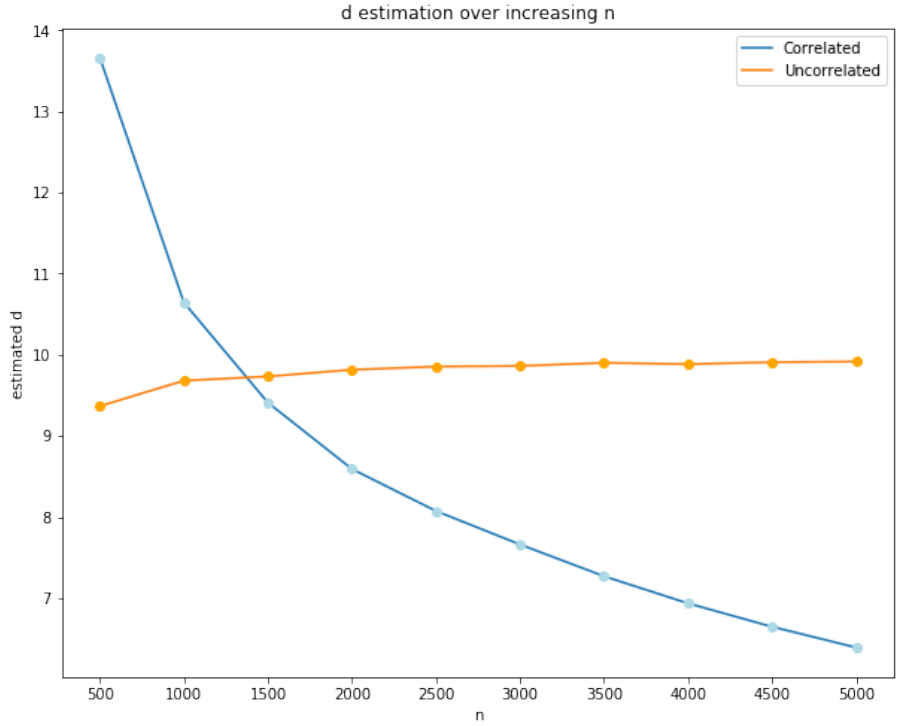


Figure 4.3: We approximate the estimated feature dimension d as we increase the sample size n over a range 500-5000. The responses are generated linearly, and we estimate d for the uncorrelated and correlated case based on formula 4.2.

In Figure 4.3, we simulate the setting for this correlated case and for an uncorrelated case where the features are completely independent. The responses are generated linearly with a dense coefficient vector, and we approximate the estimated feature dimension d as we increase the sample size n over a range 500-5000.

Figure 4.3 illustrates that, by increasing the sample size, each model converges to the number of useful features. The uncorrelated case is estimated with $d = 10$ features, while the correlated case with only $d = 6$ features. Therefore, this method of effective feature size estimation is particularly useful in settings with correlated features, in order to reduce a complex model to its effective dimensions.

Motivated by this application, we continue to investigate FIC and its feature selection properties in more experiments that measure how FIC behaves with different sample sizes and levels of model complexity, and evaluate its feature selection and predictive performances, compared to those of AIC and BIC.

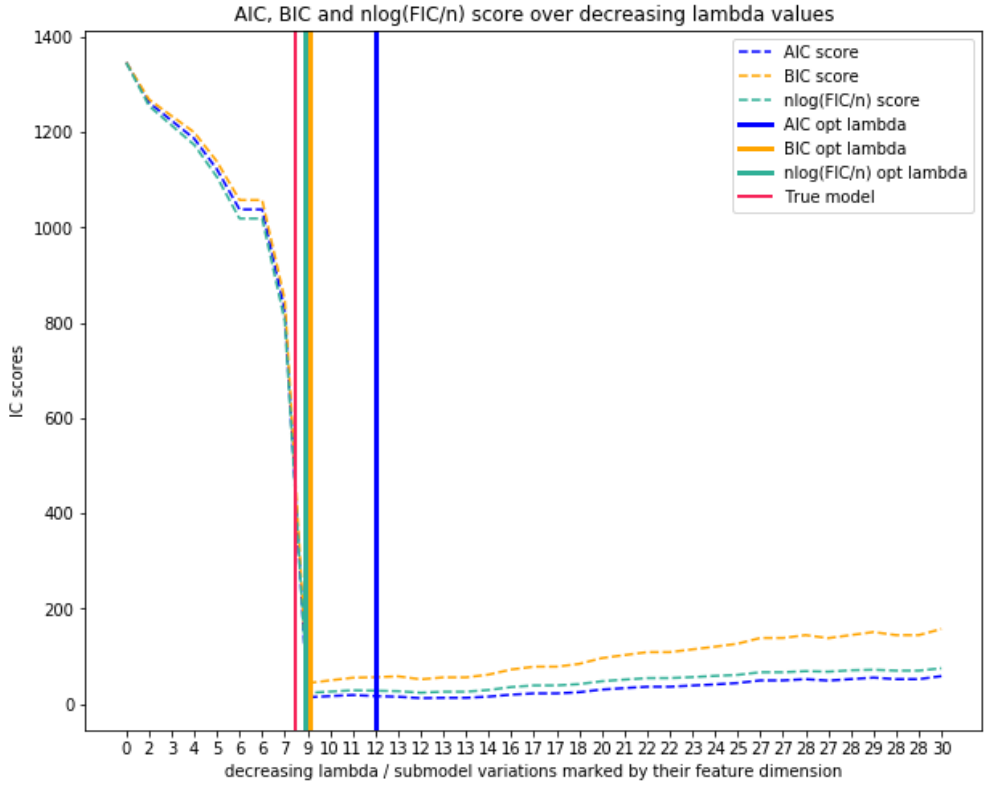
Feature selection with information criteria

Having discussed how these criteria behave with different sample sizes and feature dimensions, we continue with evaluating them for different submodels of the same full model. Our main research question in this experiment is to investigate how these criteria behave in a setting with correlated features and compare how each criterion estimates the feature dimension of the true model. We generate different submodels of the same full model with correlated features and compare which submodel is selected by each information criterion. In the next section, we compare these results by evaluating the feature selection and predictive performances of these selected submodels.

Finally, in Figure 4.4 we present the main results of our research. We show the IC estimations and the submodels each criterion selects as the one that approximates the true model best. We also compare them to the true model used to generate the data. The generative model is derived as a product of the sparse coefficient vector used to generate the responses with the design matrix of the full model. In this simulation we have used sample size $n = 200$, feature dimension of $2d = 30$, 40% sparsity in the coefficient vector of the generative model, and there are in total 20 different instances repeated in the data. We use LASSO to produce the submodels, using 15 different λ values uniformly distributed in the range $(0, 1)$.

The number of repetitions of each instance essentially means that the first d instance features are repetitions of the same 20 instance sequences, whereas the target features are not correlated and get arbitrary sequences of attributes. The number of instances that are repeated in the setting describes the correlation of the features. The larger the number of instances described in the features, the less correlated the features are. As we have seen earlier, we can use nlog(FIC/n) in order to express FIC in a scale comparable to that of BIC. Therefore, in Figure 4.4, we compare the three criteria while using nlog(FIC/n), instead of FIC. We show the selected submodel for all the criteria using horizontal axes that point to the dimension of the selected submodel. The horizontal lines of FIC and BIC are visualized side by side, however they both point to a submodel of $d = 9$.

Figure 4.4 illustrates the behavior of the three criteria in a linear regression setting with correlated features. We can compare them to the true model responsible for generating the responses. Due to the sparse coefficient vector, the dimension of active features in the true model is $d = 8$. We can observe that AIC chooses a much more complex model than the true model. Finally, FIC and BIC actually provide an estimate very close to the true model dimension. In the next section, we compare how accurate the feature selection is for the submodel each criterion proposes, and also review the predictive performances of each proposed submodel.




```
Optimal submodel dimensions:
AIC d =  12
BIC d =  9
FIC d =  9

True model d =  8
```

Figure 4.4: Estimated AIC, BIC and nlog(FIC/n) scores for different submodels. The estimations are made over decreasing λ values used for the LASSO. Instead of the λ values, we show the increasing submodel feature dimension in the x axis. The vertical lines show the selected submodel dimension chosen by each criterion and the dimension of the true submodel. The vertiacal lines for FIC and BIC are visualized side by side, showing that they both have selected a submodel with dimension 9.

Feature selection and predictive performance

In this section we compare the competence of each criterion to perform feature selection compared to the true model. Furthermore, in order to provide some further insights, we evaluate the feature selection of each criterion over different levels of sparsity used in the coefficient vector of the generative model. Finally, we compare the predictive performances of the submodels proposed by each criterion.

In order to evaluate the feature selection by each criterion, we investigate how close is the feature structure of the selected submodel to that of the true model responsible for generating the responses. We choose to do that by reviewing the precision and recall of the chosen features, and estimate an F1 score as a metric of overall comparison.

Recall is formulated by dividing the number of correct features selected by the total number of active features in the true model:

$$\text{Recall} = \frac{\text{Number of features selected correctly}}{\text{Total number of active features in true model}} .$$

Precision can be estimated by the number of correct features over the total number of features in the selected submodel:

$$\text{Precision} = \frac{\text{Number of features selected correctly}}{\text{Total number of selected features}} .$$

Finally, the F1 score is an estimation of the harmonic mean of precision and recall, formulated as:

$$\text{F1} = \frac{2\ \text{Precision}\ \text{Recall}}{\text{Precision} + \text{Recall}} .$$

We conduct the following experiment for estimating the precision, recall and F1 score for the submodel selected by each criterion compared to the generative model. We use sample size $n = 1000$, $d = 10$, density level of 70% in the coefficient vector of the generative model and 100 instances are repeated in the features. Then we run this experiment for $K = 1000$ simulations to ensure that our results are reliable. Finally, we present the estimated results averaging over all K simulated settings.

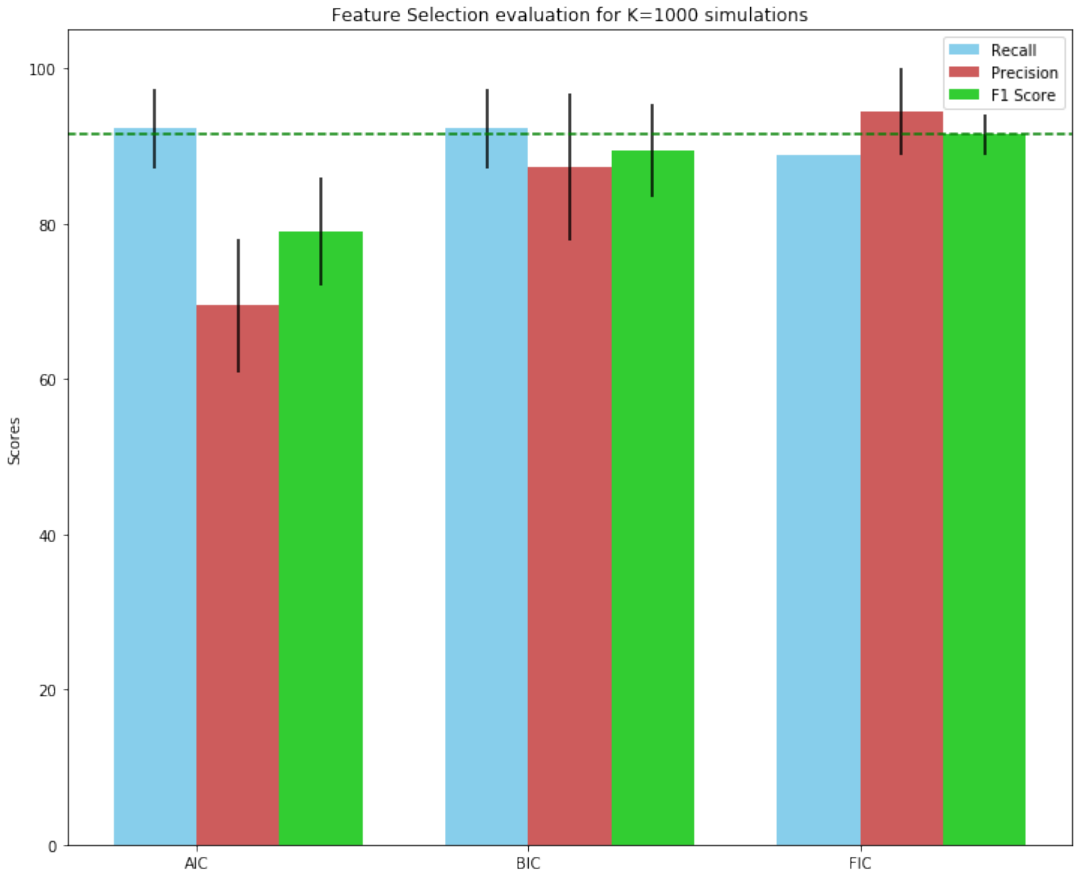


Figure 4.5: Feature selection evaluation for the three submodels selected by each criterion, measured by precision, recall and F1 score. We use sample size $n = 1000$, $d = 10$, density level of 70% in the coefficient vector of the generative model and 100 instances are repeated in the features.

We can observe that FIC yields the best F1 score, providing the most accurate option for feature selection and estimating the generative model. BIC also performs well, but AIC performs rather poorly compared to FIC and BIC. BIC is still worse than FIC, and we observe that FIC yields the best scores and smallest standard deviation over the repetitions of the experiment.

In the next experiment, we review the effect that the sparsity of the coefficient vector has on the feature selection each criterion performs. Because of FIC's primarily use for feature selection, we expect that FIC will be able to handle better estimates for sparser coefficient vectors. In the following experiment, we have used $n = 1000$, $d = 10$, we have each instance repeated twice and evaluate the F1 score for 7 different density levels ranging from 40% to 100% density. Density here refers to the percentage of non-zero elements in the coefficient vector of the generative model.

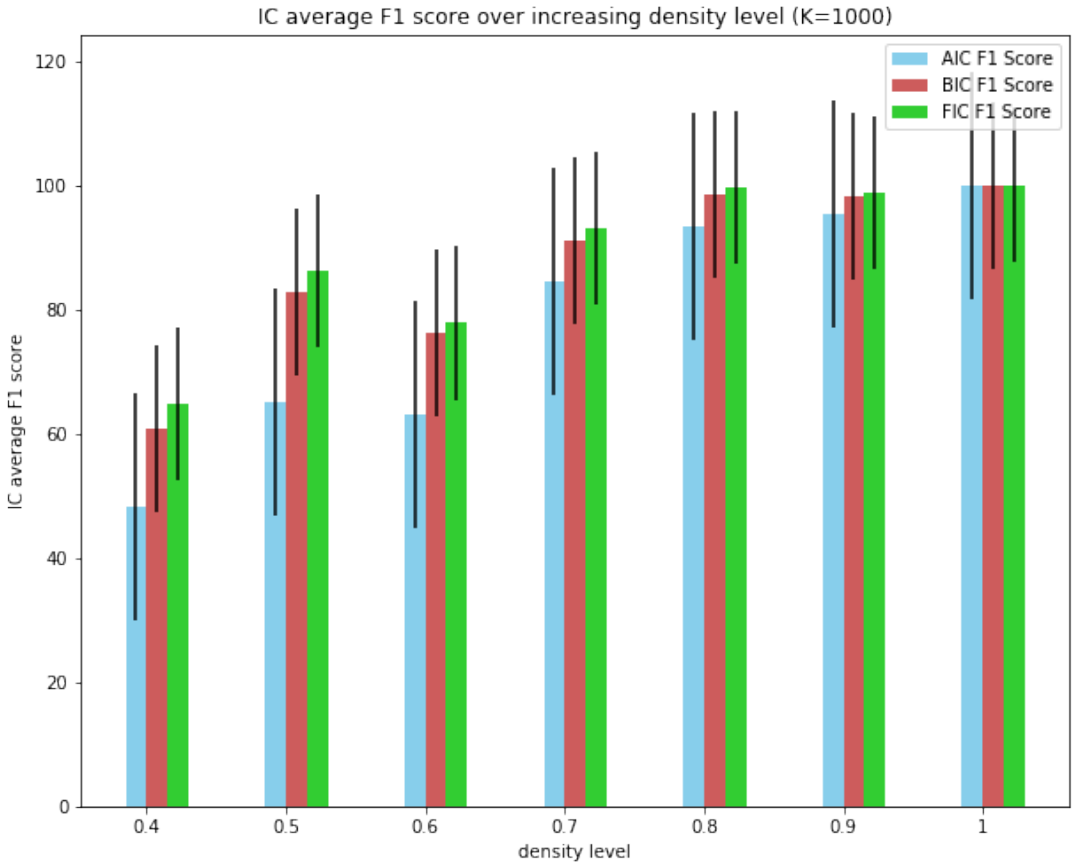


Figure 4.6: Feature selection evaluation for the three criteria measured by F1 score over different levels of density of the coefficient vector used to generate the data. Here we have used $n = 1000$, $d = 10$, we have each instance repeated twice and evaluate the F1 score for 7 different density levels ranging from 40% to 100% density.

In Figure 4.6, we see that FIC indeed yields the highest F1 scores amongst the sparser coefficient vectors, selecting better submodels than AIC and BIC. We observe that all criteria get better F1 scores as the density level increases, since they need to approximate a true model that consists of more active features. In the case of density=1, all the features of the true model are active and FIC doesn't hold the advantage of its feature selection properties anymore, which result in all the models estimating the true model equally well.

Finally, in order to estimate the predictive performance of these submodels, we are using the metric of mean squared error (MSE), which is one of the most common ways to measure error in linear regression models. We are fitting each selected submodel, and evaluate the MSE for each. MSE is formulated as:

$$\text{MSE} = \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{n} \ .$$

In the above formulation, n is the sample size, $\hat{y}_i$ describes each predicted response, while y_i is the real value of this response.

In Figure 4.7, we present the MSE for each selected submodel, using the same setting structure as in the feature selection evaluation and averaging over $K = 100$ iterations. We are using sample size $n = 1000$, $d = 10$, density level of 60% in the coefficient vector of the generative model and 5 unique instances are repeated in the features. Furthermore and we use a validation set of 30% of the full sample size. Figure 4.7 illustrates that, after averaging over the K iterations of the experiment, both AIC and BIC yield a proportionally larger error, whereas FIC's error is significantly less. However, for all the criteria the MSE is close to 1, which essentially means that all criteria have approximated very closely the generative model, and they mostly get errors in the sphere of the variance of the error used to produce the responses linearly.

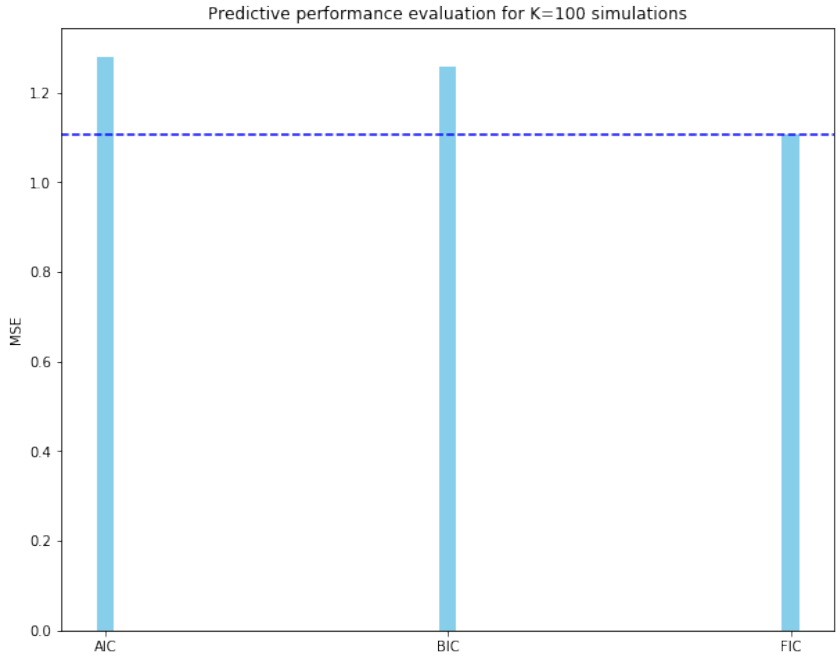


Figure 4.7: Predictive performance evaluation for the three submodels selected by each criterion, measured by MSE. We use sample size $n = 1000$, $d = 10$, density level of 60% in the coefficient vector of the generative model and 5 unique instances are repeated in the features. We use a validation set of 30% of the full sample size. We repeated the whole experiment $K = 100$ and averaged over the results.

5. Conclusions

The main research question that motivated us, was to investigate how popular information criteria like AIC and BIC work on settings with correlated data. Since, AIC and BIC penalize model complexity solely by feature dimension, we compare them to FIC which penalizes for the amount useful statistical information in the features. We evaluated the models by how accurately they could select the features of the true model, using precision, recall and F1 score as the evaluation metrics. Also, we evaluated the selected models' predictive performance using MSE. Our final results confirmed our initial intuition, since FIC was able to detect the active features of the true model, outperforming AIC and BIC.

Furthermore, we derived a formulation of FIC that allows us to approximate the effective feature dimension of a model with correlated features, under some specific setting assumptions. We compared the results of this FIC derivation on a setting with correlated features and a setting with independent features. This comparison showed that dependencies amongst features could be detected and, given sufficient sample data, the effective feature dimension was correctly estimated by the derived formula.

We were able to confirm our initial intuition on the research questions posed. We are optimistic that we can continue to further investigate model selection from this dyadic setting perspective.

Bibliography

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):717–723, 1974.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

K.P. Burnham and D.R. Anderson. *Model Selection and Inference: A practival Information Theoretic Approach*. Springer, 2002.

S. Danafar, K. Fukumizu, and F. Gomez. Kernel based information criterion. *CCSE*, 8 (1), 2014.

Paulo C. Emiliano, Mario J.F. Vivanco, and Fortunatio S. de Menezes. Information criteria: How do they behave in different models? *Computational Statistics and Data Analysis*, 69:141–153, 2014.

R.A. Fisher. *Contributions to mathematical statistics*. Wiley, 1950.

P.D. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.

Alexander Ly, Maarten Marsman, Josine Verhagen, Raoul Grasman, and Eric-Jan Wagenmakers. *A tutorial on Fisher information*. MIT Press, 2017.

Tapio Pahikkala, Antti Airola, Sami Pietilä, Sushil Shakyawar, Agnieszka Szwajda, Jing Tang, and Tero Aittokallio. Towards more realistic drug-target interaction predictions. *Briefings in Bioinformatics*, 16(2):325–337, 2014a.

Tapio Pahikkala, Michiel Stock, Antti Airola, Tero Aittokallio, Bernard De Baets, and Willem Waegeman. A two step learning approach for solving full and almost full cold start problems in dyadic prediction. *ECML/PKDD*, 2014b.

John O. Rawlings, Sastry G. Pantula, and David A. Dickey. *Applied Regression Analysis*. Springer, 1998.

Jorma Rissanen. A predictive least squares principle. *IMA J. Math. Control Inform.*, 3: 211–222, 1986.

Y. Sakamoto, M. Ishiguro, and G. Kitagawa. *Akaike Information Criterion Statistics*. KTK Scientific Publishers, 1986.

G. Schwarz. Estimating the dimensional of a model. *Annals of Statistics*, 6(2):461–464, 1978.

M. Stone. An asymptotic equivalence of choice of model by cross validation and akaike's criterion. *Journal of the Royal Statistical Society*, 39(1), 1977.

Masashi Sugiyama and Hidemitsu Ogawa. Subspace information criterion for model selection. *Neural Computation*, 13:1863–1889, 2001.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society*, 58(1):267–288, 1996.

Willem Waegeman, Krzysztof Dembczynski, and Eyke Hüllermeier. Multi-target prediction: A unifying view on problems and methods. *Data Mining and Knowledge Discovery*, 33(2):293–324, 2019.

C.Z. Wei. On predictive least squares principles. *The Annals of Statistics*, 20(1):1–42, 1992.

R. Zhang. *Model selection techniques for kernel based regression analysis using information complexity measure and genetic algorithms*. 2007.